```
from itertools import accumulate
from bisect import bisect

def knapsack(one,W,i,n1,n2,r):
  if i==n1-1:
    if one[i][1]<=W:
      return max(one[i][0]+fracKnapsack(r,W-one[i][1],n2),fracKnapsack(r,W,n2))
    else:
      return fracKnapsack(r,W,n2)
  else:
    q = -1
    q = max(one[i][0]+fracKnapsack(r,W-one[i][1],n2),one[i][0]+knapsack(one,W-one[i]
[1],i+1,n1,n2,r),knapsack(one,W,i+1,n1,n2,r),fracKnapsack(r,W,n2),q)
  return q



def fracKnapsack(r, W, n):

    vl , wt = [i[0] for i in r],[i[1] for i in r]
    acc=list(accumulate(wt))
    k = bisect(acc,W)
    return 0 if k == 0 else sum(vl[:k])+(W-acc[k-1])*(vl[k])/(wt[k]) if k!=n else sum(vl[:k])


n = int(input('Enter the number of items : '))

vl = list(map(int,input('Enter the values(space separated) : ').split()))

wt = list(map(int,input('Enter the weights(space separated) : ').split()))

op = input('Enter the type of item(space separated)[f for fractional and 0 for 0-1] : ')

W = int(input('Enter the capacity of the knapsack : '))

one = [(vl[i],wt[i]) for i in range(n) if op[i]!='f']

f = [(vl[i],wt[i]) for i in range(n) if op[i]=='f']

f = list(sorted(f, key=lambda x:x[0]/x[1],reverse=True))

print('\nThe maximum value in the knapsack is :',knapsack(one,W,0,len(one),len(f),f))
```